# An analysis of multi-objective evolutionary algorithms for training ensemble models based on different performance measures in software effort estimation

Minku, Leandro; Yao, Xin

*DOI:*
10.1145/2499393.2499396

# An Analysis of Multi-objective Evolutionary Algorithms for Training Ensemble Models Based on Different Performance Measures in Software Effort Estimation

Leandro L. Minku, Xin Yao
CERCIA, School of Computer Science, The University of Birmingham
Edgbaston, Birmingham B15 2TT, UK
{l.l.minku, x.yao}@cs.bham.ac.uk

## ABSTRACT

**Background**: Previous work showed that Multi-objective Evolutionary Algorithms (MOEAs) can be used for training ensembles of learning machines for Software Effort Estimation (SEE) by optimising different performance measures concurrently. Optimisation based on three measures (LSD, MMRE and PRED(25)) was analysed and led to promising results in terms of performance on these and other measures.
**Aims**: (a) It is not known how well ensembles trained on other measures would behave for SEE, and whether training on certain measures would improve performance particularly on these measures. (b) It is also not known whether it is best to include all SEE models created by the MOEA into the ensemble, or solely the models with the best training performance in terms of each measure being optimised. Investigating (a) and (b) is the aim of this work.
**Method**: MOEAs were used to train ensembles by optimising four different sets of performance measures, involving a total of nine different measures. The performance of all ensembles was then compared based on all these nine performance measures. Ensembles composed of different sets of models generated by the MOEAs were also compared.
**Results**: (a) Ensembles trained on LSD, MMRE and PRED (25) obtained the best results in terms of most performance measures, being considered more successful than the others. Optimising certain performance measures did not necessarily lead to the best test performance on these particular measures probably due to overfitting. (b) There was no inherent advantage in using ensembles composed of all the SEE models generated by the MOEA in comparison to using solely the best SEE model according to each measure separately.
**Conclusions**: Care must be taken to prevent overfitting on the performance measures being optimised. Our results suggest that concurrently optimising LSD, MMRE and PRED (25) promoted more ensemble diversity than other combinations of measures, and hence performed best. Low diversity is more likely to lead to overfitting.


*PROMISE* '13, October 9, 2013, Baltimore, MD, USA
ACM 978-1-4503-2016-0/13/10.
http://dx.doi.org/10.1145/2499393.2499396

## Categories and Subject Descriptors

D.2.9 [**Software Engineering**]: Management—*Cost estimation*; I.2.6 [**Artificial Intelligence**]: Learning—*Concept learning*; I.2.8 [**Artificial Intelligence**]: Problem Solving, Control Methods, and Search

## General Terms

Experimentation, Algorithms, Management

## Keywords

Software effort estimation, ensembles of learning machines, multi-objective evolutionary algorithms

## Acronyms

**Performance Measures**

| | |
|---|---|
| Corr | – Pearson Correlation |
| LSD | – Logarithmic Standard Deviation |
| MAE | – Mean Absolute Error |
| MdAE | – Median Absolute Error |
| MdMRE | – Median Magnitude of the Relative Error |
| MMRE | – Mean Magnitude of the Relative Error |
| PRED(25) | – Percentage of predictions within 25% of the actual values |
| RMSE | – Root Mean Square Error |
| StdDev | – Standard Deviation of the absolute errors |

**Algorithms**

| | |
|---|---|
| Bagging+RTs | – Bagging ensembles using RTs as the base models |
| EBA | – Estimation By Analogy |
| HaD-MOEA | – Harmonic Distance MOEA |
| MLP | – MultiLayer Perceptron |
| MOEA | – Multi-objective Evolutionary Algorithm |
| RBF | – Radial Basis Function network |
| RT | – Regression Tree |

**Others**

| | |
|---|---|
| ISBSG | – International Software Benchmarking Standards Group |
| PROMISE | – PRedictOr Models In Software Engineering Software |
| SEE | – Software Effort Estimation |

# 1. INTRODUCTION

Software Effort Estimation (SEE) is the process of estimating the effort required to develop a software project. It is a task of strategic importance in project management, as software effort is the major contributing factor for software cost. Both over and underestimations of effort can cause serious problems to a company. For instance, overestimations may result in a company losing contracts or wasting resources, whereas underestimations may result in poor quality, delayed or unfinished softwares. Due to the importance of this task, many automated methods for software cost or effort estimation have been proposed, including several machine learning approaches [32].

Recently, ensembles of learning machines started to attract the attention of the software prediction community. Ensembles are sets of prediction models trained to perform the same task and combined with the aim of improving predictive performance [6]. One of the keys for an ensemble to perform well is for its base models[1] to be diverse [4, 18], i.e., to make different errors on the same data points. This behaviour matches intuition: if models that make the same mistakes are combined into an ensemble, the ensemble will make the same mistakes as the individual models and its performance will be no better than the individual performances. On the contrary, ensembles composed of diverse models can compensate the mistakes of certain models through the correct predictions performed by the others.

Ensembles have been shown to present competitive performance for SEE in comparison to several other existing approaches [17, 16, 20, 22]. In particular, a fully automated ensemble approach has been proposed by viewing SEE as a multi-objective learning problem where the performance measures that we are interested in can be seen as objectives to be optimised [22]. A Multi-objective Evolutionary Algorithm (MOEA) is used to create SEE models trained explicitly on these measures, referred to as objective performance measures in this paper. The best models according to each objective performance measure are then combined into an ensemble referred to as Pareto ensemble.

Three objective performance measures were used in that work [22] because they behave very differently from each other, allowing the MOEA to produce diverse Pareto ensembles. These measures were: Logarithmic Standard Deviation (LSD), Mean Magnitude of the Relative Error (MMRE) and percentage of predictions within 25% of the actual values (PRED (25)). The Pareto ensembles were able to outperform several other learning machines, including state-of-the-art approaches [22], in terms of several performance measures (e.g., Mean Absolute Error (MAE)), and not only measures used as objectives.

The multi-objective ensemble approach proposed in that work [22] could be used with any objective performance measure. Using other objective performance measures would cause the MOEA to look for SEE models that specifically optimise these measures. However, it is not known how successful ensembles trained on other objective performance measures would be for SEE in comparison to LSD, MMRE and PRED(25), and whether training on these other measures would improve test performance on these measures themselves. The first aim of this work is to investigate these two points. This analysis will reveal important points to be considered when deciding which objective performance measures to use with the MOEA.

The MOEA used in [22] produces several SEE models which represent several different trade-offs among the objective performance measures, whereas only the best model according to each objective performance measure was used to compose the ensemble. As the models produced by the MOEA represent several different trade-offs among measures, an ensemble composed of all of them may achieve good performance. It is not known whether it is best to use all SEE models created by the MOEA to compose the ensemble, or solely the models with the best training performance in terms of each measure separately. Investigating this point is the second aim of this work.

This paper is organised as follows. Section 2 presents background work on machine learning for SEE. Section 3 presents the MOEA approach to create SEE ensembles. Section 4 presents the data sets used in this study. Section 5 presents the empirical study to deal with the first aim of this work. Section 6 presents the empirical study to deal with the second aim. Section 7 presents the threats to validity. Section 8 presents the conclusions and future work.

# 2. MACHINE LEARNING FOR SEE

Several different methods for automating software cost/effort estimation have been proposed [14]. For example, Shepperd and Schofield (1997) present a landmark study using Estimation by Analogy (EBA), a type of case-based reasoning based on the k-Nearest Neighbour [1] algorithm. It presents competitive performance in terms of MAE against approaches that have been more recently used for SEE, such as Regression Trees (RTs). Nevertheless, when it is not among the best approaches for a certain data set it can perform considerably worse than the best in terms of MAE [20].

Several other learning machines have been used for SEE. Examples are linear regression based on log transformed data, MultiLayer Perceptrons (MLPs), Radial Basis Function networks (RBFs) and Regression Trees (RTs) [14, 7, 20]. In particular, ensemble learning approaches have been shown to frequently outperform other approaches [17, 16, 22, 20]. An example are bagging ensembles using RTs as the base models (Bagging+RTs). A study [20] using a total of thirteen data sets and eight approaches including MLPs, RBFs, RTs and EBA showed that Bagging+RTs were frequently the best ranked approaches in terms of MAE and, when they were not ranked best, they rarely performed considerably worse than the best approach for a given data set.

Another example of ensemble approach was proposed by Kocaguneli et al. (2012) [16]. Their method combines several types of so called *solo-methods* (combinations of single learners and preprocessing techniques) to perform SEE. They reported that the ensemble presents less instability than solo-methods when ranked in terms of the total number of *wins*, *losses* and $wins - losses$ considering several different performance measures and twenty data sets. They also reported that the ensembles obtained less *losses* than other methods. As an additional contribution, their extensive study showed that the non-linear approaches CART (a type of RT) and EBA based on log transformed data can outperform other methods such as linear regression based on log transformed data. However, their approach has high implementation complexity and is not fully automated. It

[1]The prediction models that compose an ensemble are called base models.

requires an extensive experimentation procedure using several types of single learners and preprocessing techniques for creating the ensemble. It consists of selecting the "best" solo-methods in terms of *losses* and stability to compose the ensemble, by manually/visually checking and comparing their stability. The manual/visual checking process is needed because it is necessary not only to determine what solo-methods have the lowest number of *losses* (that by itself could be automated), but also to check whether these are the same as the ones comparatively more stable and what level of stability should be considered as comparatively superior.

Search based software engineering is very related to the construction of prediction models [13]. For instance, single-objective genetic programming has been applied for SEE [9, 10, 27]. SEE was then innovatively proposed to be viewed as a multi-objective learning problem where the performance measures that we are interested in can be seen as separate objectives to be optimised by a MOEA in [21, 22], and soon after in [24, 11, 25]. Search based software engineering has been used to create SEE ensemble models only very recently [22]. In this work, a MOEA is used to create SEE models trained explicitly on different objective performance measures. The best models according to each objective performance measure are then combined into an ensemble, called the *Pareto ensemble*. The following objective performance measures were used: LSD, MMRE and PRED(25). These measures were shown to behave very differently from each other, being able to create diverse ensembles. A study using a total of thirteen data sets and ten approaches (including approaches that have shown to perform well in previous studies such as Bagging+RTs, RTs and EBA based on log transformed data) showed that very good results can be obtained by using this fully automated approach. The Pareto ensemble was more frequently ranked first in terms of several performance measures, and, when it was not ranked first, its performance was usually not considerably worse than the best ranked approach's. This approach is explained in more detail in section 3.

# 3. USING MOEAS FOR CREATING SEE ENSEMBLES

This section presents in more detail the approach upon which this work is mainly based [22]. As briefly explained in section 2, this approach uses a MOEA to create SEE models by optimising on different performance measures. MOEAs are population-based optimisation algorithms that evolve sets of candidate solutions by optimising two or more possibly conflicting objectives. Even when we are interested in only one objective, adding more objectives instead of using a single-objective EA can help the optimisation not to get trapped in local optima [33].

In the case of SEE, the candidate solutions are SEE models. Candidate solutions are generated/evolved through evolutionary operators such as crossover and mutation in rounds called generations, where the total number of generations $g$ and the population size $ps$ are pre-defined. The evolutionary process is frequently guided by the concept of dominance, which is defined as follows. Consider a multi-objective optimisation problem consisting of $N$ objectives $f_i(x)$ to be minimized, where $x$ is a $p$ dimensional vector containing $p$ design or decision variables [29]. A solution $x^{(1)}$ *dominates* a solution $x^{(2)}$ iff:

$$f_i(x^{(1)}) \leq f_i(x^{(2)}) \ \forall i \wedge \exists i \mid f_i(x^{(1)}) < f_i(x^{(2)})$$

This concept can easily be generalized to problems involving maximization. The set of optimal solutions (non-dominated by any other solution) is called *Pareto front*. Even though the true Pareto front is very difficult to be found, a MOEA can find a set of acceptable solutions non-dominated by any other solution in the last generation.

Conventional MOEAs such as Non-dominated Sorting Genetic Algorithm II tend not to perform well as the number of objectives increases [15]. A couple of works in the domains of software product lines [26] and resources allocation in modular software systems [31] observed that such algorithms can perform bad even for three objectives. In [22], a MOEA called Harmonic Distance MOEA (HaD-MOEA) has been used to create SEE models. HaD-MOEA has shown to be able to cope with three objectives [31, 22]. We refer the reader to [31] should they wish to obtain more details on HaD-MOEA.

In order to use a MOEA to solve a certain problem, the objective functions, the representation and the evolutionary operators need to be defined. The objectives used in [22] were LSD, MMRE and PRED(25). These objectives are calculated based on how well a candidate solution (SEE model) performs on the training set. As explained in section 5.1, the present work will experiment on different sets of objective performance measures, as it is not known how successful different sets would be for SEE and whether using these other objective performance measures would improve test performance on these measures themselves.

The MOEA was used to generate MultiLayer Perceptrons (MLPs) [1] as the SEE models. The MLPs were represented by a real value vector of size $n_i \cdot (n_h+1)+n_h \cdot (n_o+1)$, where $n_i$, $n_h$ and $n_o$ are the number of inputs, hidden nodes and output nodes, respectively. The real value vector is manipulated by the HaD-MOEA to generate SEE models. Each position of the vector represents a weight or the bias of a node. The value one summed to $n_h$ and $n_o$ in the formula above represents the bias. The number of input nodes corresponds to the number of project independent variables and the number of output nodes is always one for the SEE task. The number of hidden nodes is a parameter of the approach. The use of one layer of hidden nodes gives MLPs the potential of being universal approximators among continuous functions. We refer the reader to [1] for more information on the structure of MLPs.

The crossover operator was defined as follows. Let $w^{p_1}$, $w^{p_2}$ and $w^{p_3}$ be three parents selected based on tournament, i.e., each of the parents is selected by randomly picking a predefined number of candidate solutions from the population and then selecting the best of them as a parent. One child $w^c$ is generated with pre-defined probability $P_c$ according to the following equation:

$$w^c = w^{p_1} + N(0, \sigma^2)(w^{p_2} - w^{p_3}),$$

where $w$ is the real value vector representing the individuals and $N(0, \sigma^2)$ is a random number drawn from a Gaussian distribution with mean zero and variance $\sigma^2$.

An adaptive procedure inspired by simulated annealing was used to update the variance $\sigma^2$ of the Gaussian at every generation. This procedure allows the crossover to be initially explorative and then become more exploitative. The

**Table 1: PROMISE data sets.**

| Data Set | # Projects | # Features | Min Effort | Max Effort | Avg Effort | Std Dev Effort |
|---|---|---|---|---|---|---|
| Cocomo81 (effort in person-months) | 63 | 17 | 5.9 | 11,400 | 683.53 | 1,821.51 |
| Nasa93 (effort in person-months) | 93 | 17 | 8.4 | 8,211 | 624.41 | 1,135.93 |
| Nasa (effort in person-months) | 60 | 16 | 8.4 | 3,240 | 406.41 | 656.95 |
| Sdr (effort in person-months) | 12 | 23 | 1 | 22 | 5.73 | 6.84 |
| Desharnais (effort in person-hours) | 81 | 9 | 546 | 23,940 | 5,046.31 | 4,418.77 |

variance is updated according to the following equation:

$$\sigma^2 = 2 - \left( \frac{1}{1 + e^{(\text{anneal\_time} - \text{generation})}} \right),$$

where *anneal_time* is a parameter meaning the number of generations for which the search is to be explorative, after which $\sigma^2$ decreases exponentially until reaching and keeping the value of one.

Mutation is performed elementwise with pre-defined probability $Pm$ according to the following equation:

$$w_i = w_i + N(0, 0.1),$$

where $w_i$ represents a position of the vector representing the MLP and $N(0, 0.1)$ is a random number drawn from a Gaussian distribution with mean zero and variance 0.1.

The offspring individuals receive further local training using Backpropagation [1], using learning rate ($lr$), momentum ($m$) and number of epochs ($ep$) as pre-defined parameters.

After the evolutionary process finishes, the non-dominated SEE models that obtained the best training performance in terms of each objective performance measure are selected to compose the ensemble, which is called *Pareto ensemble*. As explained in section 6.1, other methods to select SEE models to compose the ensemble are investigated in this work.

# 4. DATA SETS

The analysis presented in this paper is based on five data sets from the PRedictOr Models In Software Engineering Software (PROMISE) Repository[2] (section 4.1) and eight data sets based on the International Software Benchmarking Standards Group (ISBSG) Repository[3] Release 10 (section 4.2). The data sets were the same as the ones used in [22] and were chosen to cover a wide range of problem features, such as number of projects, types of features, countries and companies. They were preprocessed in the same way as in [22] for dealing with missing values and eliminating outliers which could hinder the training process. Details on preprocessing were not included here due to space restrictions and can be found in [22].

## 4.1 PROMISE Data

The PROMISE data sets used in this study are: coc81, nasa93, nasa, sdr and desharnais. Cocomo81 consists of the projects analysed by Boehm to introduce COCOMO [2]. Nasa93 and nasa are two data sets containing Nasa projects from the 1970s to the 1980s and from the 1980s to the 1990s, respectively. Sdr contains projects implemented in the 2000s and was collected at Bogazici University Software Engineering Research Laboratory from software development organisations in Turkey. Desharnais' projects are dated from late 1980s. Table 1 provides additional details and the next subsections explain their features, missing values and outliers.

Cocomo81, nasa93 and nasa are based on the COCOMO [2] format, containing as input features 15 cost drivers, the number of lines of code and the development type (except for nasa, which does not contain the latter feature). The actual effort in person-months is the dependent variable. Sdr is based on COCOMO II [3], containing as input features 22 cost drivers and the number of lines of code. The actual effort in person-months is the dependent variable. The data sets were processed to use the COCOMO numeric values for the cost drivers. The development type was transformed into dummy variables.

Desharnais follows an independent format, containing as input features the team experience in years, the manager experience in years, the year the project ended, the number of basic logical transactions in function points, the number of entities in the system's data model in function points, the total number of non-adjusted function points, the number of adjusted function points, the adjustment factor and the programming language. Actual effort in person-hours is the dependent variable.

**Table 2: ISBSG data – organisation types used.**

| Organisation Type | Id | # Projects |
|---|---|---|
| Financial, Property & Business Services | Org1 | 76 |
| Banking | Org2 | 32 |
| Communications | Org3 | 162 |
| Government | Org4 | 122 |
| Manufacturing; Transport & Storage | Org5 | 21 |
| Ordering | Org6 | 22 |
| Billing | Org7 | 21 |

## 4.2 ISBSG Data

The ISBSG repository contains a large body of data about completed software projects. The release 10 contains 5,052 projects, covering many different companies, several countries, organisation types, application types, etc. In order to produce reasonable SEE using ISBSG data, a set of relevant comparison projects needs to be selected. The data set was preprocessed to use projects that are compatible and do not present strong issues affecting their effort or sizes, as these are the most important variables for SEE. This preprocessing resulted in 621 projects. Details can be found in [22].

After that, with the objective of creating different subsets, the projects were grouped according to organisation type. Only the groups with at least 20 projects were maintained, following ISBSG's data set size guidelines. The resulting organisation types are shown in table 2.

Table 3 contains additional information about the subsets. As we can see, the productivity rate of different companies

[2]Accessed in 2011: `http://promisedata.org`.
[3]Accessed in 2011: `http://www.isbsg.org`.

**Table 3: ISBSG subsets.**

| Id | Unadjusted Function Points | | | | Effort | | | | Productivity | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Min | Max | Avg | Std Dev | Min | Max | Avg | Std Dev | Min | Max | Avg | Std Dev |
| Org1 | 43 | 2906 | 215.32 | 383.72 | 91 | 134211 | 4081.64 | 15951.03 | 1.2 | 75.2 | 12.71 | 12.58 |
| Org2 | 53 | 499 | 225.44 | 135.12 | 737 | 14040 | 3218.50 | 3114.34 | 4.5 | 55.1 | 15.05 | 9.94 |
| Org3 | 3 | 893 | 133.24 | 154.42 | 4 | 20164 | 2007.10 | 2665.93 | 0.3 | 43.5 | 17.37 | 9.98 |
| Org4 | 32 | 3088 | 371.41 | 394.10 | 360 | 60826 | 5970.32 | 8141.26 | 1.4 | 97.9 | 18.75 | 16.69 |
| Org5 | 17 | 13580 | 1112.19 | 2994.62 | 762 | 54620 | 8842.62 | 11715.39 | 2.2 | 52.5 | 23.38 | 14.17 |
| Org6 | 50 | 1278 | 163.41 | 255.07 | 361 | 28441 | 4855.41 | 6093.45 | 5.6 | 60.4 | 30.52 | 17.70 |
| Org7 | 51 | 615 | 160.10 | 142.88 | 867 | 19888 | 6960.19 | 5932.72 | 14.4 | 203.8 | 58.10 | 61.63 |

varies. A 7-way 1 factor Analysis of Variance was used to determine whether the mean productivity rate for all different subsets are equal or not. The factor considered was organisation type, with seven different levels representing each of the organisation types, and each level containing its corresponding projects as the observations. The analysis of variance indicates that there is statistically significant difference at the 95% confidence interval (p-value $< 2.2e{-}16$).

The ISBSG suggests that the most important criteria for estimation purposes are the functional size; the development type (new development, enhancement or re-development); the primary programming language or the language type (e.g., 3GL, 4GL); and the development platform (mainframe, midrange or PC). As development platform has more than 40% missing feature values for two organisation types, we used only functional size, development type and language type as input features. The normalised work effort in hours is the dependent variable. Due to the preprocessing, this is the actual development effort across the whole life cycle.

# 5. ENSEMBLES BASED ON DIFFERENT PERFORMANCE MEASURES

As explained in section 1, it is not known how successful Pareto ensembles trained on other sets of performance measures than LSD, MMRE and PRED(25) are for SEE, and whether training on other measures would improve performance on these other measures themselves. This section presents the experiments done to investigate that, and reveals important points to be considered when choosing objective performance measures for the MOEA.

## 5.1 Experimental Setup

Experiments were performed with Minku and Yao's MOEA [22] (section 3) by using the following sets of objective performance measures:

- {LSD, MMRE, PRED(25)}: these are the performance measures used in [22]. Logarithmic Standard Deviation (LSD) has been recommended by Foss et al (2003) [12] for identifying the true best model with reasonably high probability. Mean Magnitude of the Relative Error (MMRE) and the percentage of predictions within 25% of the true value (PRED(25)) are two performance measures that used to be popular for evaluation of software prediction systems, despite being based on the MRE, which is biased towards underestimations. These three measures were able to create diverse and competitive Pareto ensembles for SEE in terms of several performance measures [22].

- {MAE, RMSE, Correlation}: Mean Absolute Error (MAE) is a measure recommended by Shepperd and McDonnel (2012) [28] for being unbiased towards under and overestimations. Root Mean Square Error (RMSE) is a popular measure in the machine learning community which emphasizes large errors more. Measures of correlation have also been used to evaluate SEE approaches [7]. We use Pearson Correlation (Corr), which is widely used in sciences as a measure of the strength of linear dependence between two variables. In the case of SEE, the two variables are the estimated and the actual effort.

- {MAE, RMSE, Standard Deviation of AE}: besides using MAE and RMSE, this set also uses the standard deviation of the absolute errors (StdDev). It is desirable that SEE models are not only accurate, but also that their errors do not vary much for different projects. A more stable SEE model is more reliable, because it gives a better idea of the error that is likely to happen when estimating a project. For that reason, StdDev is included in this set of measures.

- {MdAE, MdMRE, RMSE}: measures based on the average are affected by outliers, i.e., a few projects with extreme error values. Besides RMSE, this set of measures includes two measures based on the median, which is less affected by outliers: Median Absolute Error (MdAE) and Median Magnitude of the Relative Error (MdMRE).

These four sets consider several different performance measures (a total of nine measures are used). They are also varied in the sense that two of the sets contain only error/accuracy measures ({LSD, MMRE, PRED(25)}, and {MdAE, MdMRE, RMSE}), other two of the sets contain non-error/accuracy measures ({MAE, RMSE, Corr}, and {MAE, RMSE, StdDev}), and one of the sets is known to be composed of measures that behave diversely ({LSD, MMRE, PRED(25)} [22]). We have considered sets composed of three performance measures because HaD-MOEA has shown to cope well with three objectives [31, 22].

Considering a set of $T$ projects, the formulae to calculate these performance measures are:

- $MMRE = \frac{1}{T}\sum_{i=1}^{T} MRE_i$, where $MRE_i = |\hat{y}_i - y_i|/y_i$; $\hat{y}_i$ is the predicted effort; and $y_i$ is the actual effort;

- $PRED(25) = \frac{1}{T}\sum_{i=1}^{T} \begin{cases} 1, & \text{if } MRE_i \leq \frac{25}{100} \\ 0, & \text{otherwise} \end{cases}$;

- $LSD = \sqrt{\frac{\sum_{i=1}^{T}\left(e_i+\frac{s^2}{2}\right)^2}{T-1}}$, where $s^2$ is an estimator of the variance of the residual $e_i$ and $e_i = \ln y_i - \ln \hat{y}_i$;
- $MAE = \frac{1}{T}\sum_{i=1}^{T}|\hat{y}_i - y_i|$;
- $RMSE = \sqrt{\frac{\sum_{i=1}^{T}(\hat{y}_i - y_i)^2}{T}}$;
- $Corr = \frac{\sum_{i=1}^{T}(\hat{y}_i-\bar{\hat{y}})(y_i-\bar{y})}{\sqrt{\sum_{i=1}^{T}(\hat{y}_i-\bar{\hat{y}})^2}\sqrt{\sum_{i=1}^{T}(y_i-\bar{y})^2}}$, where $\bar{\hat{y}}$ and $\bar{y}$ are the average predicted and average actual efforts, respectively;
- $MdAE =$ Median $\{|\hat{y}_i - y_i| \ /1 \leq i \leq T\}$;
- $MdMRE =$ Median $\{MRE_i \ /1 \leq i \leq T\}$.

PRED(25) and Corr are measures to be maximised, whereas the others are to be minimised.

Thirty runs were performed for each set of objectives on each data set described in section 4. Similarly to [22], in each run, for each data set, ten projects were randomly picked for testing and the remaining were used for the MOEA optimisation process/training of SEE models. Holdout of size ten was suggested by Menzies et al. (2006) [19] and allows the largest possible number of projects to be used for training without hindering testing. For the data set sdr, half of the projects were used for testing and half for training, due to the small size of the data set. The Pareto ensembles were evaluated based on LSD, MMRE, PRED(25), MAE, RMSE, Corr, StdDev, MdAE and MdMRE calculated on the test sets and then the median over the thirty runs was taken.

The parameters used by the MOEA (table 4) were the same as the ones used in Minku and Yao (2012) [22], as they were shown to obtain good results in comparison to several other approaches for SEE. Minku and Yao (2012) [22] chose these parameters in the following way. The population size and number of Backpropagation epochs were arbitrarily chosen to be 100 and 5 for the larger data sets ($\geq$ 60 projects), and arbitrarily reduced to 30 and 0 for the smaller data sets ($\leq$ 35 projects). These values were smaller for the smaller data sets in order to reduce overfitting. The remaining Backpropagation parameters were the ones more likely to obtain good results based on five preliminary executions of Backpropagation MLPs [20]. The probability of crossover and mutation were chosen among two possible values based on five preliminary executions for Cocomo81. The annealing time was the same as the one used in [5].

The comparison among the Pareto ensembles trained on different sets of objectives was aided by Friedman tests across multiple data sets [8]. Friedman test can also be used to provide rankings of the ensembles across data sets. We can choose whether small values are to represent better or worse ranks before running the test. In this work, we considered smaller values (e.g., ranking of 1st) as better rankings.

## 5.2 Analysis

Table 5 shows the average ranking and p-values of the Friedman tests comparing the Pareto ensembles over each performance measure. As we can see, there was statistically significant difference at the level of significance of 0.05 in terms of six performance measures: LSD, MMRE, MAE, Corr, MdAE and MdMRE. {LSD, MMRE, PRED(25)} was ranked first in terms of all performance measures where there was statistically significantly difference. {MAE, RMSE, Corr} was ranked last when evaluated in terms of all performance measures but MdMRE. In terms of MdMRE, even though its ranking was not the last, it was very similar to the last.

**Table 4: Parameter values used by the HaD-MOEA. $P_c$ was 0.8, $P_m$ was 0.05, anneal_time was $g/4$, and tournament size was 2.**

| Data Set | $lr$ | $m$ | $n_h$ | $ep$ | $g$ | $ps$ |
|---|---|---|---|---|---|---|
| Cocomo81 | 0.3 | 0.5 | 9 | 5 | 200 | 100 |
| Sdr | – | – | 9 | 0 | 20 | 30 |
| Nasa | 0.1 | 0.1 | 9 | 5 | 100 | 100 |
| Desharnais | 0.1 | 0.1 | 9 | 5 | 100 | 100 |
| Nasa93 | 0.4 | 0.5 | 5 | 5 | 20 | 100 |
| Org1 | 0.1 | 0.2 | 9 | 5 | 200 | 100 |
| Org2 | 0.1 | 0.1 | 5 | 5 | 100 | 100 |
| Org3 | – | – | 5 | 0 | 100 | 30 |
| Org4 | 0.2 | 0.3 | 9 | 5 | 200 | 100 |
| Org5 | – | – | 9 | 0 | 200 | 30 |
| Org6 | – | – | 3 | 0 | 20 | 30 |
| Org7 | – | – | 5 | 0 | 20 | 30 |
| OrgAll | 0.1 | 0.5 | 9 | 5 | 20 | 100 |

{MdAE,MdMRE, RMSE}'s ranking was frequently similar to {LSD, MMRE, PRED(25)}'s, but we can see that {LSD, MMRE, PRED(25)}'s ranking was still considerably better in terms of Corr and MdMRE. Wilcoxon statistical tests at the level of significance of 0.05 were performed to compare these Pareto ensembles on these two performance measures and confirm that there was statistically significant difference (p-values of 0.0398 and 0.0024, respectively).

From the analysis above, we can conclude that {LSD, MMRE, PRED(25)} obtained generally better results than the other Pareto ensembles when considering all performance measures at the same time. So, this type of Pareto ensemble can be considered as more successful than the others.

We can also observe that using a certain performance measure as an objective does not necessarily lead to better test performance in terms of this measure. {LSD, MMRE, PRED(25)} was ranked first in terms of several performance measures, even the ones not used as its objectives. {MAE, RMSE, Corr} was ranked last in terms of several performance measures, including performance measures used as its objectives. Using sets of objective performance measures different from {LSD, MMRE, PRED(25)} did not lead to better test performance on the measures being optimised. A probable reason for that is the fact that overfitting can happen on the measures being used as objective.

In order to provide evidence of that reason, we investigated the Friedman ranking of all Pareto ensembles based on the training performances (table 6). We can see that the approaches that optimised a certain set of performance measures became more competitive in terms of these measures on the training set. Their training ranking was either the first or very close to the first in terms of any of the performance measures that they optimised. This indicates that indeed the fact that they do not perform best in terms of test performance is probably linked to overfitting. An exception to that is {MdAE, MdMRE, RMSE}, whose ranking in terms of RMSE was considerably worse than {MAE, RMSE, Corr}'s, which also optimises RMSE. This is confirmed by a Wilcoxon test, which finds statistically significant difference in terms of RMSE between these two Pareto ensembles

**Table 5: Friedman ranking of Pareto ensembles across data sets based on test performance. Smaller ranks are better ranks. The smallest and biggest values are shown in yellow (light grey) and red (dark grey), respectively. The p-values of the Friedman tests are shown beside the measures' names and are marked with * when they represent statistically significant difference at the level of significance of 0.05. PRED(25) is written as PRED.**

| LSD – p-value = 0.0157* | | |
|---|---|---|
| Avg. Rank | Std. Dev. Rank | Objectives |
| 1.85 | 0.90 | {LSD, MMRE, PRED} |
| 3.31 | 0.95 | {MAE, RMSE, Corr} |
| 2.69 | 1.18 | {MAE, RMSE, StdDev} |
| 2.15 | 0.99 | {MdAE,MdMRE, RMSE} |
| MMRE – p-value < 0.0001* | | |
| Avg. Rank | Std. Dev. Rank | Objectives |
| 1.38 | 0.65 | {LSD, MMRE, PRED} |
| 3.46 | 0.52 | {MAE, RMSE, Corr} |
| 3.23 | 1.01 | {MAE, RMSE, StdDev} |
| 1.92 | 0.64 | {MdAE,MdMRE, RMSE} |
| PRED(25) – p-value = 0.1157 | | |
| Avg. Rank | Std. Dev. Rank | Objectives |
| 1.88 | 0.58 | {LSD, MMRE, PRED} |
| 2.96 | 0.85 | {MAE, RMSE, Corr} |
| 2.85 | 0.97 | {MAE, RMSE, StdDev} |
| 2.31 | 0.75 | {MdAE,MdMRE, RMSE} |
| MAE – p-value < 0.0001* | | |
| Avg. Rank | Std. Dev. Rank | Objectives |
| 1.62 | 0.87 | {LSD, MMRE, PRED} |
| 3.46 | 0.78 | {MAE, RMSE, Corr} |
| 3.00 | 1.08 | {MAE, RMSE, StdDev} |
| 1.92 | 0.64 | {MdAE,MdMRE, RMSE} |
| RMSE – p-value = 0.1047 | | |
| Avg. Rank | Std. Dev. Rank | Objectives |
| 2.15 | 0.90 | {LSD, MMRE, PRED} |
| 3.23 | 1.09 | {MAE, RMSE, Corr} |
| 2.46 | 1.33 | {MAE, RMSE, StdDev} |
| 2.15 | 0.90 | {MdAE,MdMRE, RMSE} |
| Corr – p-value = 0.0157* | | |
| Avg. Rank | Std. Dev. Rank | Objectives |
| 1.62 | 0.96 | {LSD, MMRE, PRED} |
| 3.15 | 1.07 | {MAE, RMSE, Corr} |
| 2.62 | 1.04 | {MAE, RMSE, StdDev} |
| 2.62 | 0.96 | {MdAE,MdMRE, RMSE} |
| StdDev – p-value = 0.1997 | | |
| Avg. Rank | Std. Dev. Rank | Objectives |
| 2.62 | 0.87 | {LSD, MMRE, PRED} |
| 3.08 | 1.19 | {MAE, RMSE, Corr} |
| 2.08 | 1.26 | {MAE, RMSE, StdDev} |
| 2.23 | 1.01 | {MdAE,MdMRE, RMSE} |
| MdAE – p-value < 0.0001* | | |
| Avg. Rank | Std. Dev. Rank | Objectives |
| 1.54 | 0.78 | {LSD, MMRE, PRED} |
| 3.38 | 0.65 | {MAE, RMSE, Corr} |
| 3.31 | 0.95 | {MAE, RMSE, StdDev} |
| 1.77 | 0.60 | {MdAE,MdMRE, RMSE} |
| MdMRE – p-value < 0.0001* | | |
| Avg. Rank | Std. Dev. Rank | Objectives |
| 1.23 | 0.44 | {LSD, MMRE, PRED} |
| 3.15 | 0.90 | {MAE, RMSE, Corr} |
| 3.23 | 0.93 | {MAE, RMSE, StdDev} |
| 2.38 | 0.87 | {MdAE,MdMRE, RMSE} |

**Table 6: Friedman ranking of Pareto ensembles across data sets based on training performance. Smaller ranks are better ranks. The smallest and biggest values are shown in yellow (light grey) and red (dark grey), respectively. The p-values of the Friedman tests are shown beside the measures' names and are marked with * when they represent statistically significant difference at the level of significance of 0.05. PRED(25) is written as PRED.**

| LSD – p-value = 0.1047 | | |
|---|---|---|
| Avg. Rank | Std. Dev. Rank | Objectives |
| 2.15 | 1.07 | {LSD, MMRE, PRED} |
| 2.77 | 1.24 | {MAE, RMSE, Corr} |
| 2.00 | 1.00 | {MAE, RMSE, StdDev} |
| 3.08 | 0.95 | {MdAE, MdMRE, RMSE} |
| MMRE – p-value = 0.3481 | | |
| Avg. Rank | Std. Dev. Rank | Objectives |
| 2.00 | 1.22 | {LSD, MMRE, PRED} |
| 2.54 | 1.05 | {MAE, RMSE, Corr} |
| 2.92 | 1.12 | {MAE, RMSE, StdDev} |
| 2.54 | 1.05 | {MdAE, MdMRE, RMSE} |
| PRED(25) – p-value = 0.0745 | | |
| Avg. Rank | Std. Dev. Rank | Objectives |
| 3.27 | 1.11 | {LSD, MMRE, PRED} |
| 2.42 | 1.02 | {MAE, RMSE, Corr} |
| 2.04 | 0.83 | {MAE, RMSE, StdDev} |
| 2.27 | 0.78 | {MdAE, MdMRE, RMSE} |
| MAE – p-value < 0.0001* | | |
| Avg. Rank | Std. Dev. Rank | Objectives |
| 3.62 | 0.96 | {LSD, MMRE, PRED} |
| 1.92 | 1.19 | {MAE, RMSE, Corr} |
| 1.69 | 0.63 | {MAE, RMSE, StdDev} |
| 2.77 | 0.44 | {MdAE, MdMRE, RMSE} |
| RMSE – p-value < 0.0001* | | |
| Avg. Rank | Std. Dev. Rank | Objectives |
| 3.92 | 0.28 | {LSD, MMRE, PRED} |
| 1.38 | 0.65 | {MAE, RMSE, Corr} |
| 1.77 | 0.60 | {MAE, RMSE, StdDev} |
| 2.92 | 0.49 | {MdAE, MdMRE, RMSE} |
| Corr – p-value < 0.0001* | | |
| Avg. Rank | Std. Dev. Rank | Objectives |
| 3.77 | 0.83 | {LSD, MMRE, PRED} |
| 1.31 | 0.63 | {MAE, RMSE, Corr} |
| 2.08 | 0.49 | {MAE, RMSE, StdDev} |
| 2.85 | 0.69 | {MdAE, MdMRE, RMSE} |
| StdDev – p-value < 0.0001* | | |
| Avg. Rank | Std. Dev. Rank | Objectives |
| 4.00 | 0.00 | {LSD, MMRE, PRED} |
| 1.46 | 0.66 | {MAE, RMSE, Corr} |
| 1.62 | 0.51 | {MAE, RMSE, StdDev} |
| 2.92 | 0.28 | {MdAE, MdMRE, RMSE} |
| MdAE – p-value = 0.2871 | | |
| Avg. Rank | Std. Dev. Rank | Objectives |
| 2.69 | 1.11 | {LSD, MMRE, PRED} |
| 2.54 | 1.27 | {MAE, RMSE, Corr} |
| 2.85 | 1.14 | {MAE, RMSE, StdDev} |
| 1.92 | 0.86 | {MdAE, MdMRE, RMSE} |
| MdMRE – p-value = 0.1095 | | |
| Avg. Rank | Std. Dev. Rank | Objectives |
| 2.92 | 1.26 | {LSD, MMRE, PRED} |
| 2.62 | 1.26 | {MAE, RMSE, Corr} |
| 2.69 | 1.03 | {MAE, RMSE, StdDev} |
| 1.77 | 0.60 | {MdAE, MdMRE, RMSE} |

at the level of significance of 0.05 (p-value = 0.0024). This exception demonstrates that the optimisation process of a certain performance measure is affected by the other performance measures being optimised.

We can also see that {LSD, MMRE, PRED(25)} was always ranked worst in terms of training performance on all measures where there is statistically significant difference (table 6), whereas it was ranked first in terms of test performance on these measures (table 5). This indicates that this Pareto ensemble suffered less from overfitting than the others. As shown by Minku and Yao (2012), LSD, MMRE and PRED(25) are performance measures that behave very differently from each other, being able to produce diverse Pareto ensembles. As diversity is known to help reducing overfitting [30, 23], even though surprising, it is reasonable that this Pareto ensemble would manage to obtain better test performance than other Pareto ensembles if these other ensembles are less diverse, even considering performance measures that it did not optimise.

So, we need to check if {LSD, MMRE, PRED(25)} indeed lead to more diversity than the other sets of objectives. The set of solutions found by the MOEA can be considered as diverse if it represents several different trade-offs in terms of the objectives[4]. However, different trade-offs would not really exist in the search space (or would be very scarce) if the objectives are highly correlated, because a solution that is good in one of them would also be good in the others. In particular, a high correlation for a certain pair of performance measures means that SEE models chosen to compose the ensemble based on these measures are likely to behave similarly to each other, and would thus produce a less diverse Pareto ensemble. So, we analyse the maximum correlation between any pair of objective performance measures for each type of Pareto ensemble to indicate whether it is more or less diverse. The correlation was computed in terms of training performance of all the non-dominated solutions of the final generation considering all data sets and runs of a given type of Pareto ensemble (table 7) . When the correlation involved a measure to be maximised and a measure to be minimised, it was multiplied by -1. In this way, larger values always indicate more similar behaviour, facilitating the analysis.

As we can see from table 7, the correlation between MAE and RMSE was very high, meaning that {MAE, RMSE, Corr} and {MAE, RMSE, StdDev} are less diverse and thus more likely to overfit. The correlation between MdAE and RMSE is smaller than the correlation between MAE and RMSE, but still considerably higher than the correlation between LSD and PRED(25). So, {LSD, MMRE, PRED(25)} is still more diverse than {MdAE, MdMRE, RMSE}. This is in line with the fact that {MdAE, MdMRE, RMSE} achieved similar or better test performances than {MAE, RMSE, Corr} and {MAE, RMSE, StdDev}, but still in some cases performed worse than {LSD, MMRE, PRED(25)}.

Based on the analysis shown in this section, diversity among the objective performance measures should be a primary consideration in forming the group of objectives. Diversity is even more important than personal preferences over certain measures, as a diverse set of objective performance measures can also improve test performance on measures that are not used as objectives, whereas using a certain performance measure as an objective does not necessarily lead to better test performance in terms of this measure.

[4]In more technical terms, it would be diverse if the non-dominated solutions form a good spread in the frontier.

**Table 7: Correlation between training performance of all the non-dominated solutions of the final generation of the MOEA, considering all data sets and runs. Correlations involving a measure to be maximised and a measure to be minimised are multiplied by -1. The maximum value for each set of objectives is in red (dark grey).**

| {LSD, MMRE, PRED(25)} | |
|---|---|
| LSD vs MMRE | 0.18 |
| MMRE vs PRED(25) | 0.33 |
| LSD vs PRED(25) | 0.55 |

| {MAE, RMSE, Corr} | |
|---|---|
| MAE vs RMSE | 1.00 |
| RMSE vs Corr | 0.07 |
| MAE vs Corr | 0.06 |

| {MAE, RMSE, StdDev} | |
|---|---|
| MAE vs RMSE | 0.97 |
| RMSE vs StdDev | 0.88 |
| MAE vs StdDev | 0.74 |

| {MdAE, MdMRE, RMSE} | |
|---|---|
| MdAE vs MdMRE | 0.12 |
| MdMRE vs RMSE | 0.09 |
| MdAE vs RMSE | 0.75 |

# 6. ENSEMBLES BASED ON DIFFERENT MODEL SELECTION APPROACHES

As explained in section 1, it is not known whether the strategy of selecting the non-dominated SEE models with the best training performance in terms of each measure being optimised to compose the Pareto ensemble is a better strategy than using all non-dominated SEE models created by the MOEA. This section presents the experiments done to investigate that. As the MOEA may cause overfitting, it might also be desirable to choose the SEE models that are the second best, instead of the first best in terms of each performance measure being optimised. Increasing the ensemble size while maintaining the trade-off in terms of the number of selected models based on each performance measure might also be helpful to improve performance. So, this section also investigates these selection methods.

## 6.1 Experimental Setup

The following methods were used to select nondominted SEE models produced in the last generation of the MOEA to compose the Pareto ensemble:

- EnsBest: Select the best SEE model according to the training performance on each measure being optimised. This is the strategy adopted by Minku and Yao (2012) [22] and in section 5.
- EnsSecBest: Select the second best SEE model according to the training performance on each measure being optimised.
- EnsFirstSecBest: Select the first and second best SEE models according to the training performance on each measure being optimised.
- EnsAll: Select all the non-dominated SEE models generated by the MOEA to composed the ensemble.

The MOEA used in this section uses {LSD, MMRE, PRED (25)} as objectives to be optimised, as these objectives were shown to perform better in section 5. The number of runs, parameters, training and testing sets, and performance measures used for evaluation were the same as in section 5.1. Friedman tests [8] were also used to aid the analysis.

**Table 8: Friedman ranking of Pareto ensembles across data sets based on test performance. Smaller ranks are better ranks. The smallest and biggest values are shown in yellow (light grey) and red (dark grey), respectively. The p-values of the Friedman tests are shown beside the measures' names and are marked with * when they represent statistically significant difference at the level of significance of 0.05. PRED(25) is written as PRED.**

| LSD – p-value = 0.0835 | | |
|---|---|---|
| 2.54 | 0.97 | EnsBest |
| 3.23 | 1.09 | EnsSecBest |
| 2.08 | 0.86 | EnsFirstSecBest |
| 2.15 | 1.28 | EnsAll |

| MMRE – p-value = 0.1424 | | |
|---|---|---|
| 1.92 | 1.44 | EnsBest |
| 3.08 | 1.12 | EnsSecBest |
| 2.62 | 0.87 | EnsFirstSecBest |
| 2.38 | 0.77 | EnsAll |

| PRED(25) – p-value = 0.5589 | | |
|---|---|---|
| 2.50 | 0.71 | EnsBest |
| 2.92 | 0.76 | EnsSecBest |
| 2.23 | 1.36 | EnsFirstSecBest |
| 2.35 | 0.88 | EnsAll |

| MAE – p-value = 0.1552 | | |
|---|---|---|
| 2.54 | 1.20 | EnsBest |
| 3.08 | 1.12 | EnsSecBest |
| 1.92 | 0.76 | EnsFirstSecBest |
| 2.46 | 1.20 | EnsAll |

| RMSE – p-value = 0.3225 | | |
|---|---|---|
| 2.62 | 1.26 | EnsBest |
| 2.77 | 1.09 | EnsSecBest |
| 2.69 | 0.95 | EnsFirstSecBest |
| 1.92 | 1.12 | EnsAll |

| Corr – p-value = 0.8423 | | |
|---|---|---|
| 2.31 | 1.32 | EnsBest |
| 2.77 | 1.09 | EnsSecBest |
| 2.46 | 0.97 | EnsFirstSecBest |
| 2.46 | 1.20 | EnsAll |

| StdDev – p-value = 0.8210 | | |
|---|---|---|
| 2.77 | 1.17 | EnsBest |
| 2.38 | 1.19 | EnsSecBest |
| 2.54 | 0.78 | EnsFirstSecBest |
| 2.31 | 1.38 | EnsAll |

| MdAE – p-value = 0.6930 | | |
|---|---|---|
| 2.54 | 0.97 | EnsBest |
| 2.85 | 1.14 | EnsSecBest |
| 2.31 | 1.03 | EnsFirstSecBest |
| 2.31 | 1.38 | EnsAll |

| MdMRE – p-value = 0.0184* | | |
|---|---|---|
| 1.92 | 1.12 | EnsBest |
| 3.31 | 0.75 | EnsSecBest |
| 2.08 | 0.86 | EnsFirstSecBest |
| 2.69 | 1.25 | EnsAll |

## 6.2 Analysis

Table 8 shows the Friedman ranking of the four types of Pareto ensembles compared. As we can see, there was no statistically significant difference at the level of significance of 0.05 in terms of all measures but MdMRE, for which EnsBest was the best ranked. So, there was no inherent advantage in using EnseSecBest, EnsFirstSecBest or EnsAll in comparison to EnsBest, unless more information about the data set is known, i.e., the best choice among these methods for selecting SEE to compose the Pareto ensemble is highly dependent on the data set in hands.

# 7. THREATS TO VALIDITY

**Internal validity:** we used Friedman and Wilcoxon tests to evaluate the statistical significance of the results. Four sets of objective performance measures were used, encompassing diverse combinations of nine different performance measures. Additional objective performance measures and combinations need to be investigated to extend the conclusions of this work further. Note that using larger sets of objectives is not necessarily good, because several of these measures may be highly correlated. The MOEA approach was run using the same parameters as [22], which were not fine tuned and are unlikely to be optimal, but have shown to obtain good results. As this approach contains several pre-defined parameters, it is natural that a software manager with no specialist knowledge on MOEAs and machine learning would run it with parameters that have been used previously in the literature. So, it is reasonable to reflect that in the present study. Investigation of other parameter choices is left as future work.

**Construct validity:** we used nine different performance measures (LSD, MMRE, PRED(25), MAE, RMSE, Corr, StdDev, MdAE and MdMRE) in the evaluation of the Pareto ensembles to ensure construct validity.

**External validity:** similarly to the number of data sets used in recent work [22, 20, 7], we used thirteen different data sets containing a large variety of projects from different organisations and countries to deal with external validity.

# 8. CONCLUSIONS

This work performed a comprehensive study of MOEAs to generate ensembles of SEE models based on four combinations of objective performance measures. One of the main conclusions is that using a certain performance measure as objective does not necessarily lead to better test performance on that measure, and that a probable reason for that is overfitting. So, choosing a diverse set of objective performance measures to use with the MOEA is even more important than making choices based on individual preferences over these measures. Diversity among the objective performance measures should be a primary consideration in forming the group of objectives.

An example of diverse set of measures able to produce ensembles that perform well in terms of these and several other measures is {LSD, MMRE, PRED(25)}. It obtained similar or better performance than using {MAE, RMSE, Corr}, {MAE, RMSE, StdDev} and {MdAE, MdMRE, RMSE} in terms of LSD, MMRE, PRED(25), MAE, RMSE, Corr, StdDev, MdAE and MdMRE. So, Pareto ensembles based on {LSD, MMRE, PRED(25)} can be considered as more successful than ensembles based on these three other sets of objective performance measures.

Our study also analysed different methods to select non-dominated SEE models generated by the MOEA to compose the Pareto ensemble. No inherent advantage in selecting models based on these different methods was observed in comparison to selecting the best model according to each objective performance measure. The best among these methods to select SEE models is highly dependent on the data.

As future work, other objective performance measures and combinations should be investigated, as well as other methods to select SEE models to compose the Pareto ensemble and other types of base models. The impact of using other MOEAs and parameters should also be analysed, and the relationship between low ensemble diversity and overfitting should be investigated further.

## 9. ACKNOWLEDGMENTS

This work was supported by EPSRC grant EP/J017515/1. Xin Yao was supported by a Royal Society Wolfson Research Merit Award.

## 10. REFERENCES

[1] C. Bishop. *Pattern Recognition and Machine Learning*. Springer, Singapore, 2006.

[2] B. Boehm. *Software Engineering Economics*. Prentice-Hall, Englewood Cliffs, NJ, 1981.

[3] B. Boehm, C. Abts, A. W. Brown, S. Chulani, B. K. Clark, E. Horowitz, R. Madachy, D. J. Reifer, and B. Steece. *Software Cost Estimation with COCOMO II*. Prentice-Hall, Englewood Cliffs, NJ, 2000.

[4] G. Brown, J. Wyatt, R. Harris, and X. Yao. Diversity creation methods: A survey and categorisation. *Information Fusion*, 6:5–20, 2005.

[5] A. Chandra and X. Yao. Ensemble learning using multi-objective evolutionary algorithms. *Journal of Math. Modelling and Algorithms*, 5(4):417–445, 2006.

[6] H. Chen and X. Yao. Regularized negative correlation learning for neural network ensembles. *TNN*, 20(12):1962–1979, 2009.

[7] K. Dejaeger, W. Verbeke, D. Martens, and B. Baesens. Data mining techniques for software effort estimation: A comparative study. *TSE*, 38(2):375–397, 2012.

[8] J. Demšar. Statistical comparisons of classifiers over multiple data sets. *JMLR*, 7:1–30, 2006.

[9] J. Dolado. A validation of the component-based method for software size estimation. *IEEE TSE*, 26:1006–1021, 2000.

[10] J. Dolado. On the problem of the software cost function. *IST*, 43:61–72, 2001.

[11] F. Ferrucci, C. Gravino, and F. Sarro. How multi-objective genetic programming is effective for software development effort estimation? In *SSBSE*, pages 274–275, September 2011.

[12] T. Foss, E. Stensrud, B. Kitchenham, and I. Myrtveit. A simulation study of the model evaluation criterion MMRE. *TSE*, 29(11):985–995, 2003.

[13] M. Harman. The relationship between search based software engineering and predictive modeling. In *PROMISE*, pages 2492–2499, 2010.

[14] M. Jorgensen and M. Shepperd. A systematic review of software development cost estimation studies. *TSE*, 33(1):33–53, 2007.

[15] V. Khare, X. Yao, and K. Deb. Performance scaling of multi-objective evolutionary algorithms. In C. M. Fonseca, P. J. Fleming, E. Zitzler, K. Deb, and L. Thiele, editors, *2nd International Conference on Evolutionary Multi-Criterion Optimization (EMO'03), Lecture Notes in Computer Science*, volume 2632, pages 376–390. Springer-Verlag, 2003.

[16] E. Kocaguneli, T. Menzies, and J. Keung. On the value of ensemble effort estimation. *TSE*, 38:1403–1416, 2012.

[17] Y. Kultur, B. Turhan, and A. Bener. Ensemble of neural networks with associative memory (ENNA) for estimating software development costs. *Knowledge-Based Systems*, 22:395–402, 2009.

[18] L. I. Kuncheva and C. J. Whitaker. Measures of diversity in classifier ensembles and their relationship with the ensemble accuracy. *Machine Learning*, 51:181–207, 2003.

[19] T. Menzies, Z. Chen, J. Hihn, and K. Lum. Selecting best practices for effort estimation. *TSE*, 32(11):883–895, 2006.

[20] L. Minku and X. Yao. Ensembles and locality: Insight on improving software effort estimation. *IST*, 55(8):1512–1528, 2013.

[21] L. L. Minku. Machine learning for software effort estimation. The 13th CREST Open Workshop – Future Internet Testing (FITTEST) & Search Based Software Engineering (SBSE), `http://crest.cs.ucl.ac.uk/cow/13/slides/presentation_leandro.pdf`, `http://crest.cs.ucl.ac.uk/cow/13/videos/M2U00270Minku.mp4`, May 2011.

[22] L. L. Minku and X. Yao. Software effort estimation as a multi-objective learning problem. *ACM TOSEM*, 2012 (to appear), final author's version available at `http://www.cs.bham.ac.uk/~minkull/publications/MinkuYaoTOSEM12.pdf`.

[23] M. Perrone and L. Cooper. *Artificial Neural Networks for Speech and Vision*, chapter When networks disagree: Ensemble methods for hybrid neural networks, pages 126–142. Chapman & Hall, 1993.

[24] F. Sarro. Genetic programming for effort estimation. The 14th CREST Open Workshop – Genetic Programming for Software Engineering), `http://crest.cs.ucl.ac.uk/cow/14/slides/Sarro_14thCrestOW.pdf`, `http://crest.cs.ucl.ac.uk/cow/14/videos/M2U00285Federica.mp4`, July 2011.

[25] F. Sarro, F. Ferrucci, and C. Gravino. Single and multi objective genetic programming for software development effort estimation. In *ACM SAC'12*, pages 1221–1226, Riva del Garda (Trento), Italy, 2012.

[26] A. S. Sayyad, T. Menzies, and H. Ammar. On the value of user preferences in search-based software engineering: A case study in software product lines. In *ICSE*, pages 492–501, 2013.

[27] Y. Shan, R. J. McKay, C. J. Lokan, and D. L. Essam. Software project effort estimation using genetic programming. In *ICCCAS & WESINO EXPO*, volume 2, pages 1108–1112, Chengdu, Sichuan, 2002.

[28] M. Shepperd and S. McDonell. Evaluating prediction systems in software project estimation. *IST*, 54(8):820–827, 2012.

[29] N. Srinivas and K. Deb. Multiobjective optimization using nondominated sorting in genetic algorithms. *Evolutionary Computation*, 2:221–248, 1994.

[30] S. Wang and X. Yao. Relationships between diversity of classification ensembles and single-class performance measures. *TKDE*, 25(1):206–219, 2013.

[31] Z. Wang, K. Tang, and X. Yao. Multi-objective approaches to optimal testing resource allocation in modular software systems. *TR*, 59(3):563–575, 2010.

[32] J. Wen, S. Li, Z. Lin, Y. Hu, and C. Huang. Systematic literature review of machine learning based software development effort estimation models. *IST*, 54:41–59, 2012.

[33] Y. Yu, X. Yao, and Z.-H. Zhou. On the approximation ability of evolutionary optimization with application to minimum set cover. *AI*, 180-181:20–33, 2012.